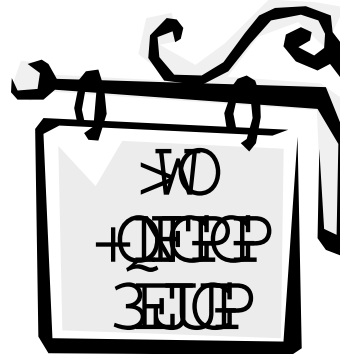


Verteilte Systeme

11. Namensdienste

Namen sind mehr als Schall und Rauch

- Eindeutige Bezeichnung von Dingen (Bezeichner)
- Anwendungsgebiete
 - Namen in einem Programm
 - Variablen, Prozeduren, Konstanten, ...
 - Rechner
 - Dienste
 - Dateien
- Funktionen eines Namens
 - Erklären
 - printer3, sturm@uni-trier.de, /de/uni-trier/fb4/cs/sturm/printer
 - Identifizieren
 - Lokalisieren
 - Auffinden und Zugriff auf Ding über den Namen



3 Funktionen \Rightarrow 3 Namenstypen

- Symbolische Namen
 - Erklärungsfunktion (für Menschen)
 - Lesbare Zeichenkette, Umfangreich
 - Aufwendiges Parsen durch den Rechner
 - Beispiel: /de/uni-trier/fb4/cs/sturm/printer
- Identifikator
 - Identifikationsfunktion
 - Für den Menschen meist "unverständlich"
 - Meist konstante Größe
 - Beispiel: 136.199.54.210, Port 80
- Adresse
 - Lokalisierungsfunktion
 - Direktes Ansprechen über Adresse möglich
 - Beispiel: 08.00.20.88.94.1e

Verteilte Systeme, Winter 2002/03

Folie 11.3

Pure und Impure

- Pure Names
 - Enthalten keine Ortsinformation
 - Nur auf Gleichheit testbar
 - Beispiele?
 - Vor- und Nachteile?
- Impure Name
 - Enthält Ortsinformation
 - Orts- oder Adreßanteil wird meist bei der Lokalisierung verwendet
 - Gültigkeit der Ortsinformation?
 - Beispiele?
 - Vor- und Nachteile?

Verteilte Systeme, Winter 2002/03

Folie 11.4

Vor- und Nachteile

Pure Names

- Vorteile
 - Häufig Erklärungsfunktion
 - Unabhängig vom Objektort
 - Erleichtert Migration
 - Zusätzliche Indirektionsstufe
 - Delegation möglich
- Nachteile
 - Direkte Lokalisierung des Objekts nicht möglich
 - Zusätzliche Indirektionsstufe
 - Mehrkosten

Impure Names

- Vorteile
 - Direkte Lokalisierung des Objekts bedingt möglich
 - Schnelle Lokalisierung
- Nachteile
 - Adreßinformation erschwert Objektmigration
 - Delegation problematisch
 - Häufig wenig verständlich

Pure Adresse !?

- Beispiele
 - IP-Adresse
 - Ethernet-Adresse
- Lokalisierungsverfahren
 - Einstufig
 - Broadcast
 - Gesuchte Station meldet sich
 - Mehrstufig
 - Routing
 - Gesuchtes Objekt bekannt
 - Weitergabe an übergeordnete Instanz
- Caching der "Weginformation"
- Bindung Name - Adresse



Unterschied zwischen Name und Adresse

- Im Prinzip keiner
- Abhängig vom gewählten Abstraktionsgrad
- Name
 - Direktes Ansprechen des gewünschten Objekts nicht möglich
 - Bindung an eine Adresse
 - 2 sichtbare Schritte
 - 1. Ermittlung der Adresse über einen Namensdienst
 - 2. Kontaktieren des gewünschten Objekts
- Adresse
 - Direktes Ansprechen des gewünschten Objekts tatsächlich (z.B. Speicheradresse) oder scheinbar (z.B. IP-Adresse) möglich
 - Unterste sichtbare Stufe
 - Kontaktieren des gewünschten Objekts

Verteilte Systeme, Winter 2002/03

Folie 11.7

Zeitpunkt der Namensbindung

- Statische Bindung
 - Beim Übersetzen, Binden (vor dem Ausführen)
 - Beispiele
 - Variablen- oder Prozedurname wird an Adresse gebunden
 - Feste Portnummern für NFS, Portmapper, ...
- Halbdynamisches Binden
 - Zum Startzeitpunkt
 - Beispiele
 - Dynamische Bibliotheken (DLLs, Shared Libs, ...)
 - Eingeben der Serveradresse beim Client
- Dynamisches Binden
 - Spätestens unmittelbar vor dem Zugriff auf das Objekt
 - Typisch für verteilte Systeme
 - Namensdienst

Verteilte Systeme, Winter 2002/03

Folie 11.8

Namensräume

- Eindeutigkeit relativ zu einem Namensraum
- Triviallösung
 - Flacher Namensraum (1)
 - Beispiele?
- Status Quo
 - Namensräume haben ebenfalls Namen
 - Hierarchischer Namensraum
 - Impliziert meist auch Architektur eines verteilten Namensdiensts

The diagram illustrates hierarchical namespaces. At the top level, there are 'Adventure' and 'Action' namespaces, each with an empty box below it. Below these is the 'PC-Spiele' namespace. Under 'PC-Spiele' is the 'Länder' namespace, which contains two sub-namespaces: 'Deutschland' and 'USA'. 'Deutschland' lists cities: Trier, Frankfurt, München, Hintertupfingen, Koblenz, and Hamburg. 'USA' lists cities: New York, Frankfurt, Boston, and Los Angeles. At the bottom level, there are 'England' and 'Spanien' namespaces, each with an empty box below it.

Folie 11.9

Architektur Aspekte

- Zentraler Namensdienst
 - Schlechte Skalierbarkeit
 - Leistungsgpaß
 - Ausfallsicherheit
- Skalierbarkeit
 - Verteilter Namensdienst
 - Hierarchische Struktur
- Leistungsgpaß
 - Datenreplikation
- Ausfallsicherheit
 - Datenreplikation
- Datenkonsistenz
 - "Fast" Read-Only-Daten
 - Inkonsistenz unkritisch
 - Nochmal Auskunft bei "falsch verbunden"

The diagram shows a hierarchical tree structure of nameservers. At the root is a 'Nameserver' box. It branches into two 'Nameserver' boxes. The left one branches into three 'Nameserver' boxes, and the right one branches into two 'Nameserver' boxes. Each of these five boxes further branches into two 'Nameserver' boxes, resulting in a total of 10 leaf nodes at the bottom level.

Folie 11.10

Funktionen eines Namensdiensts

- Server verwaltet ein oder mehrere Namensräume
- Clientseitiges API
 - **Insert (Name)**
 - **Delete (Name)**
 - Nicht immer direkt zugreifbar
 - **Bind (Name1, Name2)**
 - **Unbind (Name1, Name2)**
 - **Query (Name1) returns Name2**
- Serverseitiges API
 - **Resolve (Name)**
 - Anfrage bei einem übergeordneten Server
- Schutz- und Sicherheitsaspekte
 - Wer darf welche Informationen abfragen?
 - Wer darf welche Informationen hinzufügen/ändern?

Verteilte Systeme, Winter 2002/03

Folie 11.11

Erweiterter Funktionsumfang

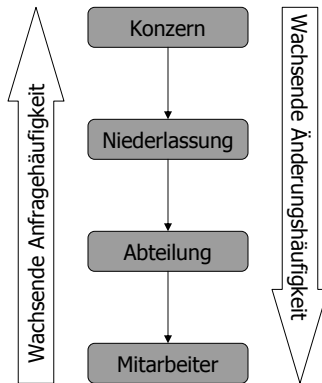
- Directory Services
- Speicherung von "beliebigen" Attributen zu einem Namen
- Beispiele
 - Email-Directory
 - Name, Adresse, Fähigkeiten, ...
 - Personaleinträge
 - Name, Adresse, Email, Gehalt, ...
 - ...
- Wachsende Verbreitung
 - X.500, LDAP, Active Directory (Microsoft), ...
- Erweiterte Queries
 - Pattern Matching
 - Filter



Verteilte Systeme, Winter 2002/03

Folie 11.12

Replikation



- Leistungssteigerung und Ausfallsicherheit
- Änderungen bei Namensräumen seltener als bei Namen
 - Namensräume sind Ordnungsprinzip
- Lesende Anfragen häufiger als Änderungen
- Temporäre Inkonsistenzen tolerierbar
 - Effiziente Replikat-"Konsistenz"
 - Caching akzeptabel
- Obere Ebenen besonders gut replizierbar
 - Hohe Anfragehäufigkeit
 - Geringe Änderungshäufigkeit

Verteilte Systeme, Winter 2002/03

Folie 11.13

Replikatkonsistenz

- Änderungszeitpunkt
 - Periodische Aktualisierung
 - Inkonsistenzfenster tolerieren
 - Wie würde man vorgehen?
 - Jede Änderung sofort "pushen"
 - Kleines Inkonsistenzfenster
 - Hohe Server- und Netzbelastung
 - Abhängigkeit von der Ebene der geänderten Daten?
- Ausgangspunkt
 - Primärserver
 - Alle Änderungen auf einem zentralen Server
 - Serverausfall, Leistungengpaß
 - Beispiel: NIS
 - Symmetrische Lösung
 - Verteilter Abgleich



Verteilte Systeme, Winter 2002/03

Folie 11.14

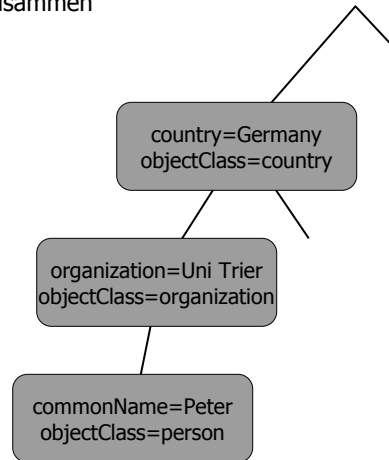
11.1 Symbolischer Namensdienst Beispiel: LDAP

Übersicht LDAP

- Basiert auf X.500
 - Bekannter ISO-Standard eines Directory Service
 - Volles ISO-OSI-Protokoll (Aufwand)
 - ASN.1-Notation bei verwendeten Datenstrukturen
 - Zugangsprotokoll: Directory Access Protocol (DAP)
- Lightweight DAP = LDAP (Aktuelle Version 2)
 - Direkte TCP-Adaption
 - Stringrepräsentation (Heterogene Systeme)
 - Protokoll für Anfrage an LDAP-Server
 - LDAP \subset DAP: Anfrage an X.500-Server ebenfalls realisierbar
- Standards
 - RFC 1777: LDAP-Standard
 - RFC 1778: Stringrepräsentation für Attribute
 - RFC 1779: Stringrepräsentation für Namen
 - RFC 1558: Syntax für Suchfilter

Einträge im Directory Service

- Einträge setzen sich aus Attributen zusammen
- Jedes Attribut besteht aus
 - Typ
 - Ein oder mehreren Werten
- Attributwerte sind Strings
- Jeder Eintrag besitzt spezielles Attribut **objectClass**
 - Legt Attributtypen fest (Namen der Attribute)
 - Welche Attribute benötigen zwingend Werte?
 - Welche Attribute sind optional?



Verteilte Systeme, Winter 2002/03

Folie 11.17

Distinguished Names

- Relative Distinguished Name (RDN)
 - Relativ zum gemeinsamen Parent eindeutiger Name
 - Besteht aus den Werten ein oder mehrerer Attribute
 - Höchstens ein Wert pro Attribut
- Distinguished Name (DN)
 - Eindeutige und absolute Objektbezeichnung
 - Alle RDNs von der Wurzel zum Objekt
- X.500-Namen basieren auf ASN.1
- LDAP-Namen
 - Stringrepräsentation
 - Beispiel
 - "co=Germany, O=Uni Trier, cn=Peter"

Verteilte Systeme, Winter 2002/03

Folie 11.18

Protokollaufbau

- LDAP-Server
 - TCP-Port 389
- Schritte
 - 1. Aufbau der TCP-Verbindung
 - 2. Authentifizierung
 - 3. LDAP-Operationen
 - 4. Kommunikation beenden
- Operationen (X)
 - **S**earch (Suche Einträge)
 - **M**odify (Attribute modifizieren)
 - **A**dd (Neuen Eintrag aufnehmen)
 - **D**el (Eintrag löschen)
 - **M**odifyRDN (Letzten Namensteil ändern)
 - **C**ompareDN (Bedingung über Attribute testen)
 - **A**bandonRequest (Ausstehende Operation abbrechen)

```

sequenceDiagram
    participant Client
    participant LDAP-Server
    Client->>LDAP-Server: BindRequest
    LDAP-Server-->>Client: BindResponse
    Client->>LDAP-Server: XRequest
    LDAP-Server-->>Client: XResponse
    Note over Client,LDAP-Server: ...
    Client->>LDAP-Server: UnbindRequest
    
```

Folie 11.19

LDAP-API (RFC 1823)

```

sequenceDiagram
    participant Client
    participant LDAP-Server
    Client->>LDAP-Server: ldap_X_s(Req)
    LDAP-Server-->>Client: Result
    
```

```

sequenceDiagram
    participant Client
    participant LDAP-Server
    Client->>LDAP-Server: ldap_X(Req)
    LDAP-Server-->>Client: Result
    Note over Client: ldap_result()
    
```

- Grob eine Funktion pro Protokollnachricht
- Synchron und Asynchrone Operationsaufrufe
 - Synchron: Blockade des Clients bis zum Eintreffen des Resultats
 - Asynchron: Keine Blockade; Client fragt Resultat explizit ab
- Asynchrone Operationen können abgebrochen werden
- Authentisierungsvarianten
 - Simple
 - Klartext-Passwort
 - Kerberos Version 4 (LDAP)
 - Kerberos Version 4 (X.500)

Folie 11.20

Anfrage

```
int ldap_search_s (
    LDAP *ld,
    char *base,
    int scope,
    char *filter,
    char *attrs[],
    bool attrsonly,
    LDAPMessage *result
)
```

- q Verbindung zum LDAP-Server
 - Handle `ld`
- q Scope (LDAP_SCOPE_...)
 - **BASE**
 - **ONELEVEL**
 - **SUBTREE**
- q Suchfilter `filter`
- q Welche Attribute und eventuell Attributwerte sollen zurückgegeben werden (`attrs`)
- q Nur Attribute oder auch Werte (`attrsonly`)?
- q Variante mit Timeout
 - `ldap_search_st()`

Verteilte Systeme, Winter 2002/03

Folie 11.21

Angabe von Filtern

```
<filter> ::= '(' <filtercomp> ')'  
<filtercomp> ::= <and> | <or> | <not> | <item>  
<and> ::= '&' <filterlist>  
<or> ::= '|' <filterlist>  
<not> ::= '!' <filter>  
<filterlist> ::= <filter> | <filter> <filterlist>  
<item> ::= <simple> | <present> | <substring>  
<simple> ::= <attr> <filtertype> <value>  
<filtertype> ::= <equal> | <approx> | <greater> | <less>  
<equal> ::= '='  
<approx> ::= '~='  
<greater> ::= '>='  
<less> ::= '<='  
<present> ::= <attr> '=*'  
<substring> ::= <attr> '=' <initial> <any> <final>  
<initial> ::= NULL | <value>  
<any> ::= '*' <starval>  
<starval> ::= NULL | <value> '*' <starval>  
<final> ::= NULL | <value>
```

Verteilte Systeme, Winter 2002/03

Folie 11.22

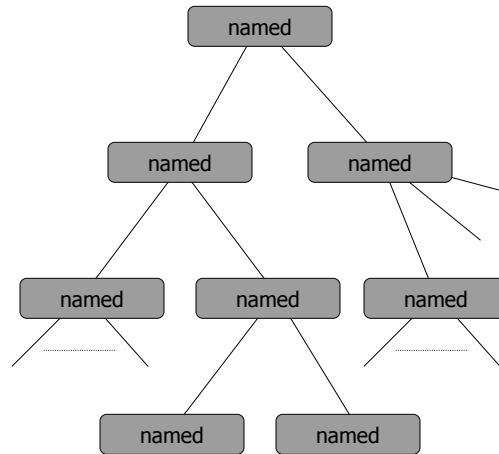
11.2 Lokalisierungsdienst Beispiel: DNS

Übersicht DNS

- Domain Name Service
- Hierarchischer Namensraum: balvenie.uni-trier.de
- Transformation
 - Domain Name → IP-Adresse (Lookup)
 - IP-Adresse → Domain Name (Reverse Lookup)
- UNIX-API
 - `gethostbyname ()`
 - `gethostbyaddr ()`
- IETF-Standard
 - RFC 1034 (Konzepte)
 - RFC 1035 (Implementierungsdetails)
- Gängige Implementierung
 - Berkeley Internet Name Domain Server (BIND)

Organisation der DNS-Server

- Hierarchische Struktur
- Ausfallsicherheit
 - Primary Name Server
 - Secondaries
- Administrative Einheiten
 - Zone



Verteilte Systeme, Winter 2002/03

Folie 11.25

Einträge

- Ca. 20 verschiedene Eintragsformen
 - Viele nicht mehr gültig
- Gängige „Record“-Einträge
 - A: IP-Adresse
 - NS: NS-Server
 - CNAME: Kanonischer Name
 - PTR: Pointer-Record
 - HINFO: Host-Information
 - MX: Mail Exchange Record
- Pointer-Query
 - Gegeben IP-Adresse; Gesucht: Kanonischer Name
 - `in-addr.arpa` Domain

Verteilte Systeme, Winter 2002/03

Folie 11.26